**Cursor官方指南：如何在Cursor中處理大型代碼庫？（附個人實踐解讀）**

**https://www.youtube.com/watch?v=HVSLDJZ3Iss**

最近發現 Cursor 的官方文檔多了個 Guide 分類，其中有一篇主題為“**如何在 Cursor 中處理大型代碼庫**”的指南，內容有一定啟發性，所以分享給大家。

：[https://docs.cursor.com/guides/advanced/large-codebases](https://link.zhihu.com/?target=https%3A//docs.cursor.com/guides/advanced/large-codebases)

一張含有 文字, 螢幕擷取畫面 的圖片

AI 產生的內容可能不正確。

以下是這篇指南的中文翻譯 & 我的實踐解讀（*解讀部分會用斜體展示*）：

相比處理小型專案，處理大型代碼庫會遇到更多的新挑戰。借鑒我們從 Cursor 自身代碼庫的擴展經驗 & 客戶管理大型代碼庫的洞察，我們發現了處理複雜代碼庫的有效模式。

在這份指南中，我們將介紹一些我們認為對處理大型代碼庫行之有效的方法。

一張含有 文字, 螢幕擷取畫面, 行, 字型 的圖片

AI 產生的內容可能不正確。

**一、使用 Chat 快速熟悉陌生代碼**

如果你是第一次接觸大型代碼庫，可能會比較有挑戰。

相比用grep、搜索和點擊來找到你需要的代碼庫特定部分。我們更建議你使用 Chat，你可以直接向 Cursor 提問來找到你的目標內容，並獲得關於它是如何工作的詳細解釋。

*注：1、這裡的 Chat 並不是 v0.45及更早版本的 Chat 模式，而且泛指和 Cursor 的對話，包括當前的 Ask、Agent、Manual；*

*2、grep 是一個經典的命令列工具，用於在文本中搜索指定的字串或模式。*

在下面這條視頻中，我們正在尋求 Cursor 説明以找到代碼庫索引的實現細節，同時讓它給出一些示例來幫助我們更好地理解。

[https://mintlify.s3.us-west-1.amazonaws.com/cursor/images/guides/advanced/large-codebases/qa.mp4](https://link.zhihu.com/?target=https%3A//mintlify.s3.us-west-1.amazonaws.com/cursor/images/guides/advanced/large-codebases/qa.mp4)

*除了用 Chat 來快速索引和理解代碼庫特定內容，我們還可以用 Chat 來快速瞭解專案結構，比如：*

請分析項目代碼庫，用[mermaid](https://zhida.zhihu.com/search?content_id=257450184&content_type=Article&match_order=1&q=mermaid&zd_token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJ6aGlkYV9zZXJ2ZXIiLCJleHAiOjE3NDc3MTQ3OTMsInEiOiJtZXJtYWlkIiwiemhpZGFfc291cmNlIjoiZW50aXR5IiwiY29udGVudF9pZCI6MjU3NDUwMTg0LCJjb250ZW50X3R5cGUiOiJBcnRpY2xlIiwibWF0Y2hfb3JkZXIiOjEsInpkX3Rva2VuIjpudWxsfQ.yTCOUvknBHyX0VjQvYl6vFDri1ns9xP7MKJGMOSkfAU&zhida_source=entity)圖表展示專案結構，以及各種檔的依賴關係等

*不過這種方法需要結合mermaid.live、*[*http://draw.io*](https://link.zhihu.com/?target=http%3A//draw.io)*等工具來實現視覺化。*

*一張含有 文字, 螢幕擷取畫面, 軟體 的圖片

AI 產生的內容可能不正確。*

*如果你需要熟悉的專案剛好在 GitHub 上，那麼可以嘗試用 Devin 推出的*[*DeepWiki*](https://zhida.zhihu.com/search?content_id=257450184&content_type=Article&match_order=1&q=DeepWiki&zd_token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJ6aGlkYV9zZXJ2ZXIiLCJleHAiOjE3NDc3MTQ3OTMsInEiOiJEZWVwV2lraSIsInpoaWRhX3NvdXJjZSI6ImVudGl0eSIsImNvbnRlbnRfaWQiOjI1NzQ1MDE4NCwiY29udGVudF90eXBlIjoiQXJ0aWNsZSIsIm1hdGNoX29yZGVyIjoxLCJ6ZF90b2tlbiI6bnVsbH0.6VaI8WmEJGjhDiCkqzGhI6iFIhJyFe7NlSjqbyNWXEE&zhida_source=entity)*（*[*www.deepwiki.com*](https://link.zhihu.com/?target=http%3A//www.deepwiki.com/)*）來熟悉項目。之前在知識星球也介紹過：*

*DeepWiki 可以理解為 GitHub 代碼庫的維琪百科。原本的 Github 代碼庫只是簡單地展示 README.md 文檔，而 DeepWiki 會深入分析代碼結構、檔、配置，幫我們生成以下這些內容，包括：*

* *結構化的專案文檔：像維琪百科一樣，清晰地列出專案概覽、核心模組、關鍵概念等；*
* *可交互的視覺化mermaid圖表：可以直接展示各種檔的依賴關係、模組的調用關係等；*
* *智能問答：支援用自然語言直接提問關於代碼庫的任何具體問題。*

*一張含有 文字, 螢幕擷取畫面 的圖片

AI 產生的內容可能不正確。*

*DeepWiki 的使用也不麻煩：*

*1、直接在它的官網點擊或者搜索任意 GitHub 倉庫，就可以看到一個類似維琪百科的介紹頁；*

*2、或者在打開的 GitHub 倉庫，直接修改它的URL，比如*[*https://github.com/microsoft/playwright-mcp*](https://link.zhihu.com/?target=https%3A//github.com/microsoft/playwright-mcp)*，只需要將URL中的GitHub替換成deepwiki，即*[*https://deepwiki.com/microsoft/playwright-mcp*](https://link.zhihu.com/?target=https%3A//deepwiki.com/microsoft/playwright-mcp)*，同樣可以得到對應 GitHub 倉庫的一個維琪百科介紹頁。*

*如果不想跳出 Cursor 就使用 DeepWiki 能力，可以考慮 DeepWiki MCP，這是非官方推出的一個 MCP Server，它實現的能力就是通過 MCP 接收 DeepWiki 的URL，爬取所有相關頁面，將它們轉換為 Markdown 格式，並返回一個文檔或頁面清單。*

*：*[*https://github.com/regenrek/deepwiki-mcp*](https://link.zhihu.com/?target=https%3A//github.com/regenrek/deepwiki-mcp)

為了讓 Cursor 更深入地理解代碼庫結構，建議在【Cursor Settings-Features】中啟用【Include project structure】，以獲得更好的性能。

一張含有 文字, 螢幕擷取畫面, 軟體, 多媒體軟體 的圖片

AI 產生的內容可能不正確。

**二、編寫特定領域知識的規則（Rules）**

如果讓你給新協作者介紹代碼庫，你會提供哪些背景資訊以確保他們可以快速上手？

對於每個組織或專案，都存在一些隱性知識可能沒有完全記錄在你的文檔中。而**有效地使用規則，可以讓 Cursor 全面瞭解代碼庫**（包括這些隱性知識）**。**

比如，你可以編寫一條簡短的規則，來告訴 Cursor 怎麼實現新功能或服務：

---

描述：添加一個新的 [VSCode](https://zhida.zhihu.com/search?content_id=257450184&content_type=Article&match_order=1&q=VSCode&zd_token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJ6aGlkYV9zZXJ2ZXIiLCJleHAiOjE3NDc3MTQ3OTMsInEiOiJWU0NvZGUiLCJ6aGlkYV9zb3VyY2UiOiJlbnRpdHkiLCJjb250ZW50X2lkIjoyNTc0NTAxODQsImNvbnRlbnRfdHlwZSI6IkFydGljbGUiLCJtYXRjaF9vcmRlciI6MSwiemRfdG9rZW4iOm51bGx9.43yXziy118oz2wqU-P5bN1_DZGYHAvskHbiuzxxT7R0&zhida_source=entity) 前端服務

---

1. \*\*介面定義：\*\*

- 使用 `createDecorator` 定義一個新的服務介面，並確保包含 `\_serviceBrand` 以避免錯誤。

2. \*\*服務實現：\*\*

- 在一個新的 TypeScript 檔中實現該服務，擴展 `Disposable`，並使用 `registerSingleton` 將其註冊為單例。

3. \*\*服務貢獻：\*\*

- 創建一個貢獻檔來導入和載入該服務，並在主入口點註冊它。

4. \*\*上下文集成：\*\*

- 更新上下文以包含新服務，從而允許在整個應用程式中訪問。

---

又比如，你希望 Cursor 根據檔案格式來遵守相應規則，就可以使用 [Auto Attached 模式](https://zhida.zhihu.com/search?content_id=257450184&content_type=Article&match_order=1&q=Auto+Attached+%E6%A8%A1%E5%BC%8F&zd_token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJ6aGlkYV9zZXJ2ZXIiLCJleHAiOjE3NDc3MTQ3OTMsInEiOiJBdXRvIEF0dGFjaGVkIOaooeW8jyIsInpoaWRhX3NvdXJjZSI6ImVudGl0eSIsImNvbnRlbnRfaWQiOjI1NzQ1MDE4NCwiY29udGVudF90eXBlIjoiQXJ0aWNsZSIsIm1hdGNoX29yZGVyIjoxLCJ6ZF90b2tlbiI6bnVsbH0.wP2fg_rYHdI9437iBhltTgk8wFWBOcnd_gIy8tk8OXc&zhida_source=entity)來實現。

---

globs: \*.ts

---

- 使用 bun 作為包管理器。相關腳本請參考 [package.json](mdc:backend/reddit-eval-tool/package.json)

- 檔命名統一使用 kebab-case（短橫線連接的小寫命名）

- 函數名與變數名統一使用 camelCase（駝峰命名法）

- 所有硬編碼的常量使用 UPPERCASE\_SNAKE\_CASE（全大寫+底線分隔）

- 優先使用 `function foo()` 的函式定義方式，而非 `const foo = () =>`

- 使用 `Array<T>` 的形式，而不是 `T[]`

- 推薦使用具名匯出（named exports），例如 `export const variable ...` 或 `export function ...`，避免使用默認匯出（default export）

*關於Cursor Rules，我之前分享過系列文章，涵蓋最早的.cursorrules，以及最新的Project Rules：*

[*cursor教程 | 如何根據不同專案寫好一份合格的cursorrules?*](https://link.zhihu.com/?target=https%3A//mp.weixin.qq.com/s%3F__biz%3DMzIyMzk3MTEwNQ%3D%3D%26mid%3D2247484363%26idx%3D1%26sn%3D6f670f86a4994220076fd56bee3ce569%26scene%3D21%23wechat_redirect)

[*.cursorrules將被移除，大家現在就可以遷移使用Project Rules，控制代碼更精准*](https://link.zhihu.com/?target=https%3A//mp.weixin.qq.com/s%3F__biz%3DMzIyMzk3MTEwNQ%3D%3D%26mid%3D2247485250%26idx%3D1%26sn%3D60a91bb5d2938cc2896b490f4eb8fa27%26scene%3D21%23wechat_redirect)

[*Cursor Rules在實際開發中的三種層級&實際應用（附20個常用Rules）*](https://link.zhihu.com/?target=https%3A//t.zsxq.com/O7fds)

*在早期，我對Cursor Rules的理解更多是給AI立規範。*

*因為Cursor中的各種大模型具有不同的編碼特點，加之大模型的隨機性，決定了我們和大模型的每次對話，其實就相當於和一個新的協作者在結對程式設計。這就好比團隊每天會來一位新同事和你協作推進同一個專案，你需要反復給新同事介紹專案要求，這種溝通一定會把你給“逼瘋”的。所以需要Cursor Rules來給AI立規範。*

*之前的這種理解，更多是放在個體和Cursor的結對程式設計上。隨著對企業Cursor使用場景的瞭解，我發現Cursor Rules，不只是給AI的規範，也是確保團隊成員在大型專案中使用統一的AI輔助標準。*

*值得一提的是，程式設計只是Cursor的應用場景之一，還有很多人活用它作為 AI IDE 在文本編輯上的優勢，去做論文寫作、小說創作、知識庫整理等等，Cursor Rules在這些場景中就可以起到統一上下文的作用。*

**三、緊跟“計畫制定”的過程**

對於大的項目改動，前期投入足夠時間去思考並制定一個精確、合理的計畫，可以顯著提高 Cursor 的輸出品質。

當你發現提示詞改來改去，依然沒有得到自己想要的結果，不如退一步，從頭開始創建一個更詳細的計畫，就像你給同事編寫 PRD（Product Requirements Document，產品需求文檔）一樣。**通常，這裡面最困難的部分是確定應該做出哪些改動，比較適合我們人類來完成**。有了正確的指令後，我們就可以將實施的部分交給 Cursor。

當然，AI也能幫忙制定計劃。

你可以在 Cursor 中打開 Ask 模式，將你從專案管理系統、內部文檔或零散的想法中擁有的任何上下文輸入進去。想想你在代碼庫中要用到的檔和依賴項。這可能是一個包含你想要集成的代碼片段的檔，也可能是一整個專案檔案。

這是一個“讓AI幫助制定計劃”的示例prompt：

制定一個關於如何創建新功能的計畫（類似於 @existingfeature.ts）

如有任何問題，請向我提問（最多 3 個）

請務必搜索代碼庫

@Past Chats（我之前的探索提示）

以下是來自 [專案管理工具] 的更多背景資訊：

[粘貼的工單描述]

我們可以要求模型通過“向人類提問”來制定計劃並收集上下文，並且參考之前的探索提示和工單描述。建議使用類似於 [Claude-3.7-Sonnet](https://zhida.zhihu.com/search?content_id=257450184&content_type=Article&match_order=1&q=Claude-3.7-Sonnet&zd_token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJ6aGlkYV9zZXJ2ZXIiLCJleHAiOjE3NDc3MTQ3OTMsInEiOiJDbGF1ZGUtMy43LVNvbm5ldCIsInpoaWRhX3NvdXJjZSI6ImVudGl0eSIsImNvbnRlbnRfaWQiOjI1NzQ1MDE4NCwiY29udGVudF90eXBlIjoiQXJ0aWNsZSIsIm1hdGNoX29yZGVyIjoxLCJ6ZF90b2tlbiI6bnVsbH0.IRehjt_m32EtHMwE-UlF3Aec4V-BpHjwJSkzXUaRtuY&zhida_source=entity) 、[Gemini-2.5-Pro](https://zhida.zhihu.com/search?content_id=257450184&content_type=Article&match_order=1&q=Gemini-2.5-Pro&zd_token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJ6aGlkYV9zZXJ2ZXIiLCJleHAiOjE3NDc3MTQ3OTMsInEiOiJHZW1pbmktMi41LVBybyIsInpoaWRhX3NvdXJjZSI6ImVudGl0eSIsImNvbnRlbnRfaWQiOjI1NzQ1MDE4NCwiY29udGVudF90eXBlIjoiQXJ0aWNsZSIsIm1hdGNoX29yZGVyIjoxLCJ6ZF90b2tlbiI6bnVsbH0.jZ3aypyI79lgYyOn15xfsG4odo_K9qwWZkktfMANyk0&zhida_source=entity) 或 o3 這類推理模型，因為它們可以理解項目改動的意圖並更好地制定計劃。

一張含有 文字, 字型, 行, 螢幕擷取畫面 的圖片

AI 產生的內容可能不正確。

*關於在Cursor開發中大型項目，目前大家應用比較廣泛的就是Plan&Act模式：*

* *Plan模式：負責制定計劃*
* *Act模式：負責實施計畫*

*之前分享的文章*[*Cursor解決bug總在繞圈？可以嘗試引入 Memory Bank*](https://link.zhihu.com/?target=https%3A//t.zsxq.com/E1KCK)*，就是介紹怎麼在Cursor中應用Plan&Act模式來實現更高效地開發。*

---

描述：

全域設置：

始終應用：true

------

## 模式規則

你有兩種操作模式：

1、Plan 模式：你將與用戶共同制定計劃，收集所有必要資訊以進行更改，但不會實際執行任何更改

2、Act 模式：你將根據計畫對代碼庫進行實際修改

- 你預設以Plan模式開始，只有在使用者批准計畫後才會切換到Act模式

- 每次回應開頭需標明當前模式，計畫模式顯示'# 模式：Plan'，執行模式'顯示# 模式：執行'

- 除非使用者明確輸入指令要求切換至Act模式，否則你將始終保持Plan模式

- 每次回應後自動返回Plan模式，使用者輸入計畫指令時也會切換回Plan模式

- 在Plan模式下若使用者要求執行操作，你需要提醒當前處於Plan模式，需要先批准計畫

- 在Plan模式下，每次回應都必須輸出完整的更新後計畫

*當然，這種方法也可以借助Cursor的自訂模式（即Manual模式）來實現：*

*一張含有 文字, 螢幕擷取畫面, 軟體, 多媒體軟體 的圖片

AI 產生的內容可能不正確。*

*大家在【Cursor Settings-Features】中開啟開啟【Custom modes】後，就可以在對話方塊進行Plan模式和Act模式的配置，後續就可以根據自己的實際開發流程自由切換工作模式。*

*一張含有 文字, 螢幕擷取畫面, 軟體, 多媒體軟體 的圖片

AI 產生的內容可能不正確。*

*就像前面官方提到的，對於大的專案改動，前期投入足夠時間去思考並制定一個精確、合理的計畫是非常值得的。所以在Plan階段，其實還可以進一步拆解為Research模式和Plan模式：*

* *Research模式：主要做前期調研，包括需求收集、功能理解以及模擬實現*
* *Plan模式：主要對前期調研的內容轉化成具體可執行的計畫，前面提到的PRD就是計畫的一種展現形式*

*這裡簡單提下這三種模式的配置，大家可以根據自己的實際情況進行調整：*

***Research模式****：模型可以選擇Claude-3.7-Sonnet、Gemini-2.5-Pro等推理模型；tools可以按截圖所示選擇，其中MCP非必選；Advanced option的custom instructions可以複製使用下面的指令：*

*一張含有 文字, 螢幕擷取畫面, 軟體, 多媒體軟體 的圖片

AI 產生的內容可能不正確。*

在提出解決方案之前，從工作空間和代碼庫的多個來源中收集全面資訊。

分析代碼和近期變更，但不得自動修復問題。

不得修改任何代碼。 如需使用代碼展示解決方案，直接在回復中以純Markdown文本格式提供。

在提供解決方案時，保留相關上下文資訊（如檔路徑、函數名或模組），以便使用者理解。

避免基於不明確的假設進行分析或建議，必要時向用戶請求澄清。

以一致的格式（如代碼塊、列表或標題）呈現分析結果和解決方案，便於用戶快速閱讀。

***Plan模式****：和Search模式的配置基本一致，區別在於Advanced option的custom instructions：*

*一張含有 文字, 螢幕擷取畫面, 軟體, 多媒體軟體 的圖片

AI 產生的內容可能不正確。*

\*\*充分研究和審查\*\*：

在開始制定計劃前，需全面研究和審查所有相關細節，包括我們討論過的內容、文檔、代碼庫和外部資源。

\*\*制定詳細實施計畫\*\*：基於研究結果，創建詳細的實施計畫，但不直接修改代碼，計畫需要包含以下內容：

- 代碼級別的變更指南，需完全基於代碼庫審查。

- 潛在風險分析及應對措施（如相容性問題、性能影響）。

- 測試策略（如單元測試、集成測試）以驗證變更效果。

\*\*使用Mermaid圖表\*\*：對於複雜流程，使用Mermaid圖表（流程圖/時序圖/狀態圖）進行說明：

- 使用清晰的標籤和節點連接。

- 不同操作類型使用顏色編碼（如輸入為藍色，處理為綠色，輸出為橙色）。

\*\*計畫檔存儲\*\*：

- 所有計劃檔必須存儲在 .plans/ 目錄下。

- 檔命名格式為 PLAN-{id}-{summary}.md：

- {id} 為 .plans/ 目錄及其子目錄中的唯一編號。

- {summary} 為任務的簡短描述。

- 檔採用 Markdown 格式，包含任務完成狀態（如 [ ] 未完成，[x] 已完成）等。

*Act模式：一般可以直接選擇Agent模式。*

**四、選擇合適的工具來完成工作**

在有效使用 Cursor 的過程中，選擇合適的工具至關重要。考慮你要達成的目標，並選擇能讓你保持流暢的工作方式的方法。

一張含有 文字, 螢幕擷取畫面, 收據, 字型 的圖片

AI 產生的內容可能不正確。

每個工具都有其優勢：

* 使用 Tab 進行快速編輯，讓你成為掌控者
* 當你需要對代碼的特定部分進行專注的修改時，Cmd + K 會是神來之筆
* 對於需要 Cursor 理解更廣泛上下文的大規模修改，Chat 模式再合適不過了

當你使用 Chat 模式（可能會感覺有點慢，但非常強大）時，通過提供良好的上下文來幫助它幫助你。使用@files 指向你想要參考的類似代碼，或使用@folder 來更好地理解你的專案結構。並且不要害怕將更大的更改分解成更小的部分——從頭開始聊天有助於保持事情專注和高效。

*這部分乍一看似乎和處理大型代碼庫沒太大關係，但如果將這部分內容放到大型專案的代碼重構中去重新理解，你就會發現它非常有實踐價值。*

*對於大型項目的代碼重構，一般沒法一次性完成，拆分（如模組化）是比較常見的做法。當拆分後，大家就可以根據每個階段、模組、步驟的特點來選擇對應的工具，比如：*

***初期規劃階段****：使用****Chat 模式分析專案結構，識別需要重構的模組或檔。結合@folder 理解全域上下文，制定重構計畫。***

***局部優化階段：使用 Cmd + K****針對特定代碼塊進行優化，如重構函數、提取公共邏輯或替換過時 API。*

***快速調整階段：使用 Tab****補全重複性代碼或修復小問題，保持編碼流暢。*

***驗證與收尾：再次使用 Chat 模式****檢查跨檔一致性，或用****Cmd + K****微調，確保重構後的代碼符合預期。*

*根據任務規模和複雜性靈活切換工具，並結合上下文管理和任務分解，我們就可以在 Cursor 中高效完成大型項目重構，最大化工具的價值。*

**總結：**

1、縮小變更範圍，不要試圖一次做太多；

2、有條件的話，對話中盡可能包含相關上下文；

3、使用 Chat、Cmd + K 和 Tab 進行它們最擅長的事情；

4、經常創建新的對話*（之前在知識星球也分享過，每次完成一個新功能，最好新開對話，避免之前的上下文對後面的功能開發造成污染）*；

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

AI 產生的內容可能不正確。

5、使用 Ask 模式制定計劃，使用 Agent 模式實施計畫。

以上就是Cursor官方處理大型代碼庫的指南以及我的實踐解決，希望對大家有啟發。

可以搭配Cursor首席設計師 Ryo Lu 分享的 Cursor 正確用法一起閱讀 >>>[Cursor官方下場談Cursor正確用法，以及我的實踐解讀](https://zhuanlan.zhihu.com/p/1900338212569854320)